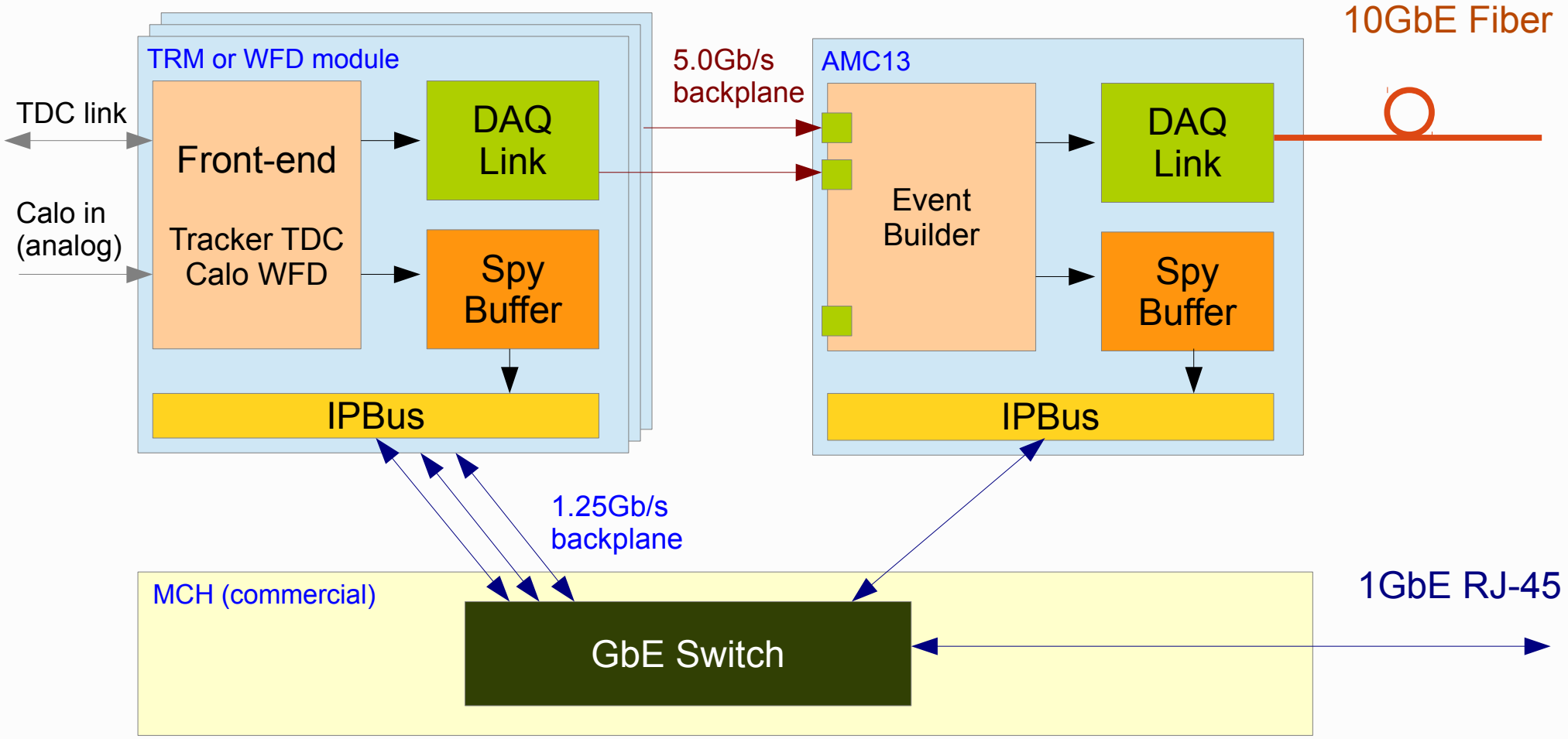
G-2 Tracker

Readout and DAQ
For 2014 testbeam

E. Hazen

Tracker / Calo uTCA Readout

(data paths in one crate)



Spy path for testing and “local DAQ” (Ethernet readout)

Fast DAQ path for data taking (automatic once initialized)

CMS-style spy buffer

- Partitioned memory; one event (spill) per partition
- One partition mapped to IPBus address space at a time
  - Tracker: (2k*32) per TDC * 24 TDCs = 48k words
  - WFD: ~ 64MB for 60 calo channels in one crate
- Read out using IPBus block transfer
- Write to control register to advance to next event

- Tracker and Calo event sizes differ by 1000,
  so we will want to use different partition sizes

Test Beam Proposal

- Implement “fake TDC” block which generates fake events (maybe already done?)
- Implement CMS-style spy buffer to manage event stream
- Develop IPBus slave(s) for readout and control

Register Outline (test beam)

Module Control

Module ID	serial number, firmware revision
Status	
Control (momentary)	reset all, reset counters, fake trigger...
Control (latched)	run mode, fake data...

Data Readout (aka "Spy buffer")

Event buffer	memory region with current event
Word count	word count in current buffer
Advance buffer	write to advance to next buffer

TDC Control

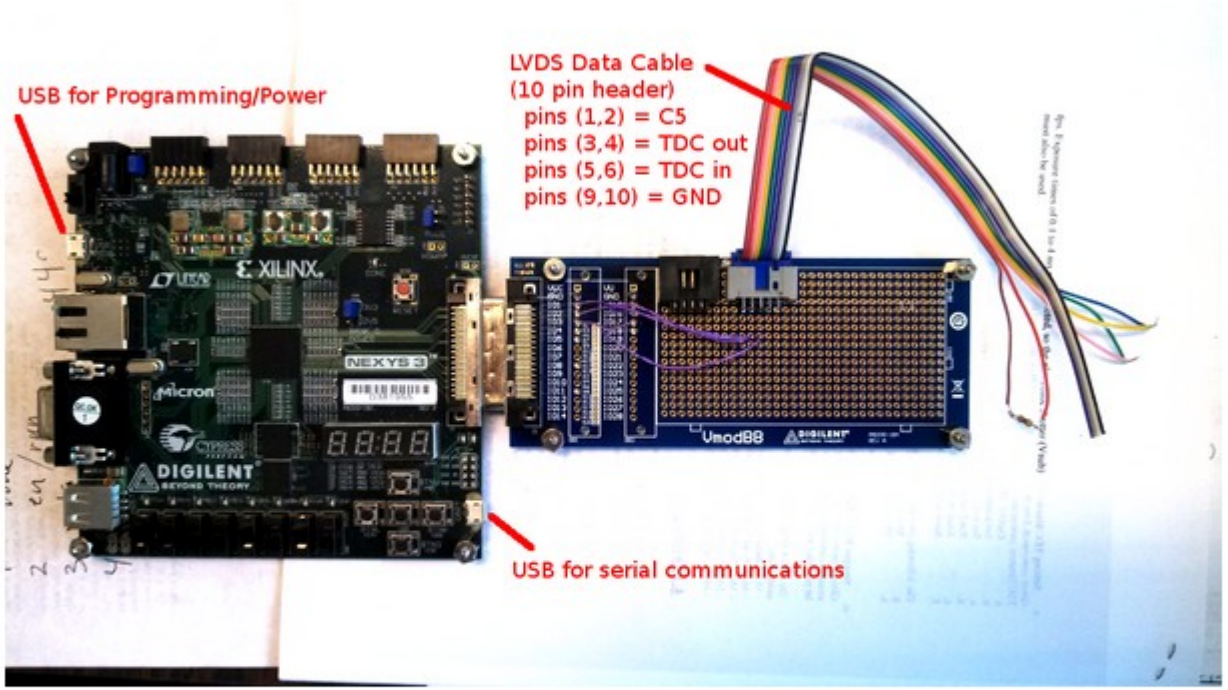
C5 command	send arbitrary c5 command
Write register	write to TDC register using C5 sequence

Monitoring

Counters	number of 8b10b K-chars received
	number of 8b10b data chars received
	number of spills received
	etc...

Test Rig Documentation

This document describes a test firmware for g-2 tracker TDC testing and readout.



This firmware initially runs on a Digilent Nexys3 board. The board may be powered by a USB micro USB cable connected to the "USB Prog" connector. (a second MicroUSB cable is needed to communicate with the board for operation, connected to the "USB UART" connector).

Clocks are derived from the 100MHz on-board oscillator. The following I/O are implemented on the VHDCI connector (adapter to 10-pin ribbon header needed):

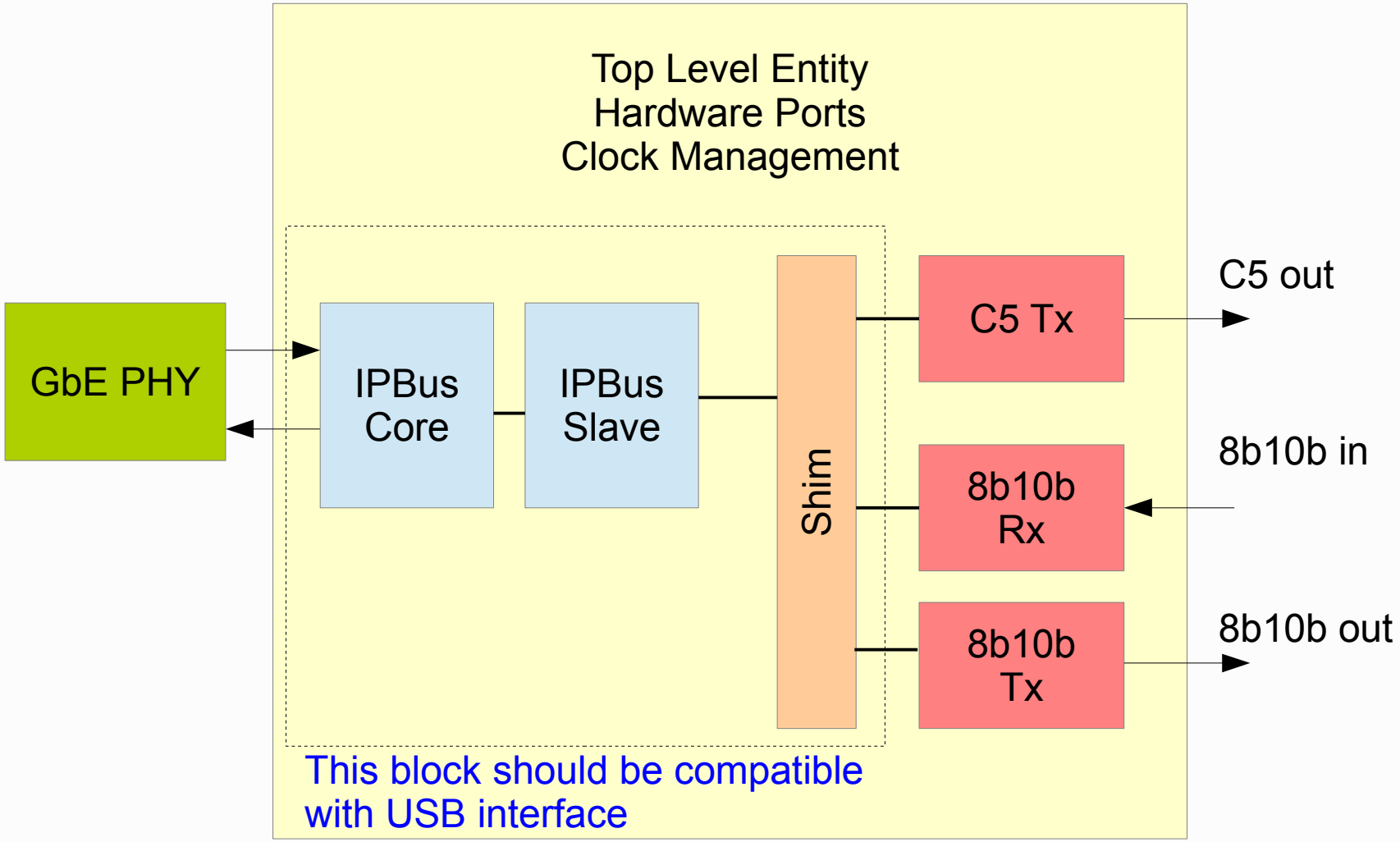
Google doc describing the Nexys3 based test board being used at Fermilab and BU

(USB interface)

Documentation here: http://goo.gl/0OtN0D

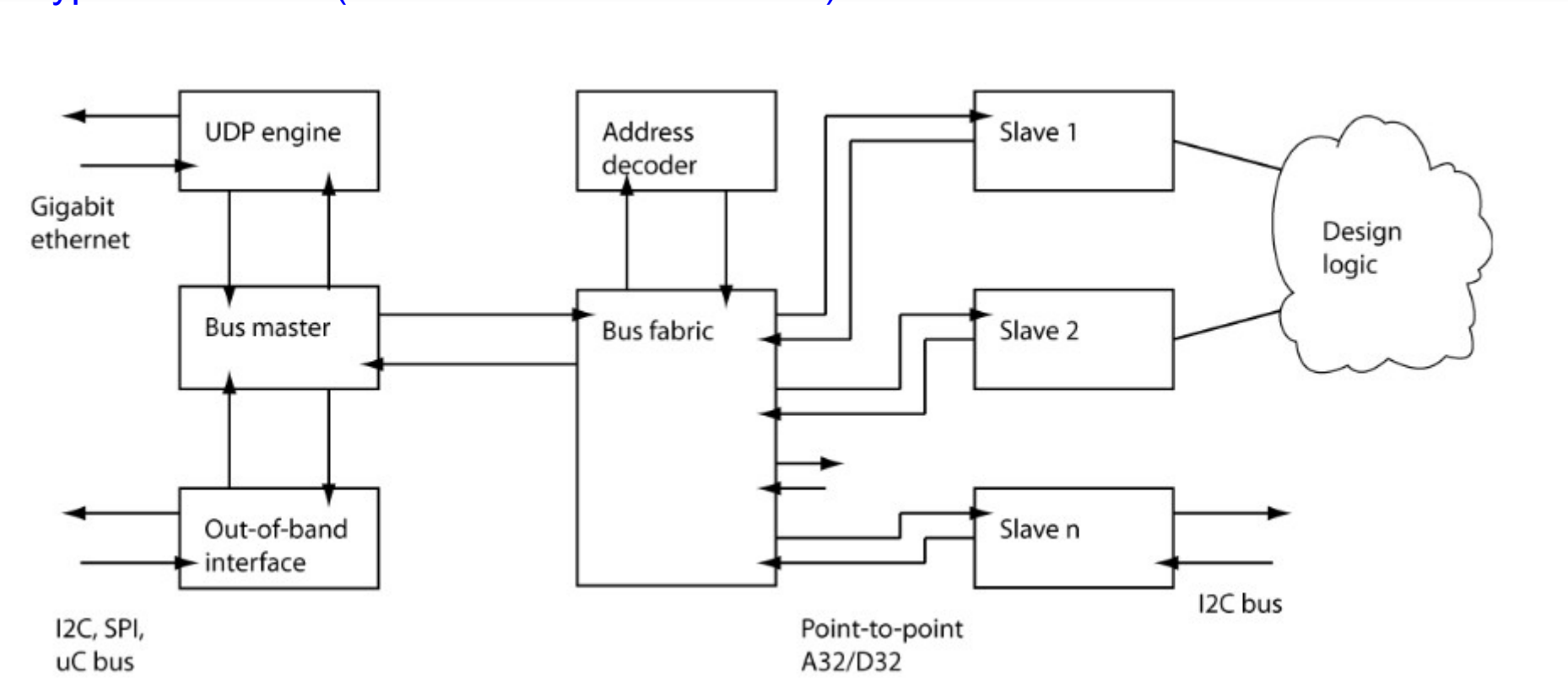
Backup

Proposed Firmware Structure



IPBus Details

Typical structure (from IPBus documentation)



C5 Transmitter Entity

Inputs all synch'd to 125MHz, 40MHz used only for output
(clock domain crossing handled inside this entity)

```
entity c5_top is

  port (
    clk125 : in  std_logic;                    -- 125MHz clock for computer interface
    clk40  : in  std_logic;                    -- 40MHz clock to run C5 output
    rst_n  : in  std_logic;                    -- active low reset
    din    : in  std_logic_vector(4 downto 0);  -- data in (0:3) plus CTRL
    c5_en  : in  std_logic;                    -- transmit enable
    c5_out : out std_logic);                   -- encoded C5 output

end entity c5_top;
```

8b10b Readout

Data recovery, K.28.5 comma detect, 8b1b0 decoder, 8k byte FIFO
This block requires only 125MHz clock
All control signals 1 clock wide synch'd to clk125 except asynch rst_n

```
entity rec_8b10b_top is

  port (
    clk125     : in  std_logic;          -- system clock
    rst_n      : in  std_logic;          -- active low reset
    serial     : in  std_logic;          -- serial data in
    data_out   : out std_logic_vector(7 downto 0);  -- output data
    fifo_full  : out std_logic;          -- fifo full flag
    fifo_empty : out std_logic;          -- fifo empty flag
    k_char     : out std_logic;          -- K char at FIFO top
    locked     : out std_logic;          -- 8b10b comma aligned
    err        : out std_logic;          -- 8b10b input error
    fifo_wr    : out std_logic;          -- fifo write output for debug
    fifo_clr   : in  std_logic;          -- fifo clear
    test       : out std_logic_vector(4 downto 0);
    fifo_rd    : in  std_logic           -- fifo read strobe
    );

end entity rec_8b10b_top;
```

Fake 8b10b Output

Generate simulated TDC data in 8b10b format
Requires 125MHz clock
Output data: header is K.28.5, 0xda, 0xca, 0xfe
Data words are 0xAnnnBnnn
where nnn is hex word number

```
entity fake_TDC is

  port (
    clk     : in  std_logic;                     -- 125MHz clock
    rst_n   : in  std_logic;                     -- asynchronous reset
    trig    : in  std_logic;                     -- trigger a fake event
    test    : out std_logic_vector(3 downto 0);  -- debug outputs
    length  : in  std_logic_vector(11 downto 0); -- length in words
    ser_out : out std_logic);                    -- 8b10b serial out

end entity fake_TDC;
```